The Interpreter Pattern

Gerald Britton

Pluralsight Author

@GeraldBritton www.linkedin.com/in/geraldbritton

Overview

- Common domain-specific languages
- Review of Backus normal form
- Define a language for scrambled eggs
- Understand the Interpreter pattern in UML
- Build an Interpreter implementation
- Review benefits and drawbacks

Domain Specific Languages

SQL – Structured Query Language

CSS – Cascading Style Sheets

HTML – Hypertext Markup Language

JSON – JavaScript Object Notation

PHP – PHP: Hypertext Preprocessor

Defining a DSL

Formal grammars in Backus normal form (BNF)

```
format_spec ::=  [[fill]align][sign][#][0][width][.precision][type]
fill        ::=  <a character other than '}'>
align       ::=  "<" | ">" | "=" | "^"
sign        ::=  "+" | "-" | " "
width       ::=  integer
precision   ::=  integer
type        ::=  "b" | "c" | "d" | "e" | "E" | "f" | "F" | "g" | "G" | "n" | "o"
| "s" | "x" | "X" | "%"
```
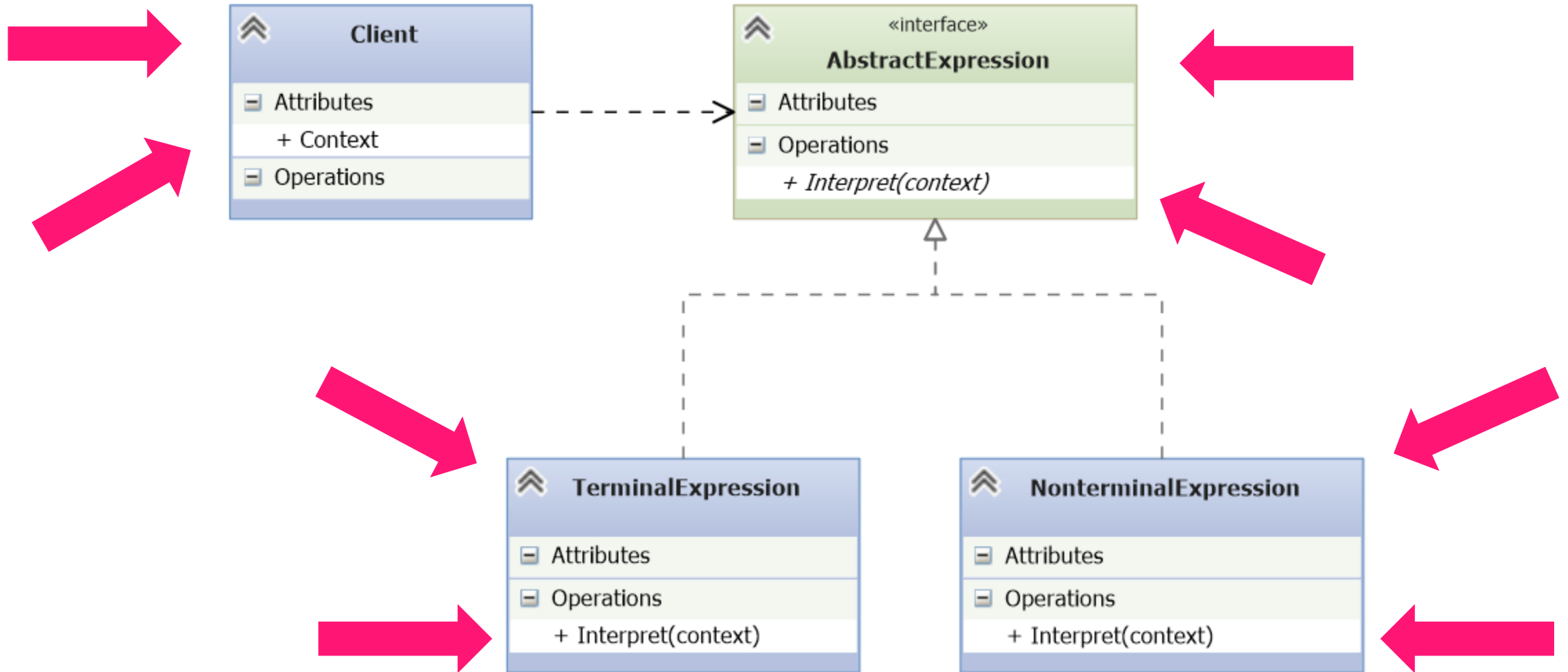
DSL for Making Scrambled Eggs

```
expression ::= command | sequence | repetition
sequence   ::= expression ";" expression
command    ::= "break egg" | "mix in bowl" | "melt butter in pan" | "cook eggs"
               | set
repetition ::= while variable expression
variable   ::= [A-Z[0-9]+]+
set        ::= set variable ("true" | "false")


Example:

break egg; break egg; mix in bowl; melt butter in pan; set NOTCOOKED true;
while NOTCOOKED cook eggs; set NOTCOOKED false
```

Interpreter Pattern Structure

Demo

Build the interpreter

Use a simple AST in the client

- Python `compile()` or `ast.parse()` functions

Can use the Visitor pattern

Use Flyweight for terminal symbols

Consequences

Benefits

Easy to extend the grammar

Easy to implement

Easy to change expression processing

Drawbacks

Complex grammars need maintenance

Summary

Interpreter is good for simple languages

Use when simplicity is important

Ideal for scripting languages

Related patterns:
- Composite
- Flyweight
- Visitor
- Iterator

Keep in your back pocket!